Available online at www.sciencedirect.com

ScienceDirect

Procedia Computer Science 62 (2015) 31 – 38

Procedia Computer Science

The 2015 International Conference on Soft Computing and Software Engineering (SCSE 2015)

Analysis of a Monte Carlo Tree Search in Knight-Amazons

Hikari Kato[a], Mayumi Takaya[a], Akihiro Yamamura[a,*]

[a]*Department of Computer Science and Engineering, Akita University*

Abstract

We study whether or not a Monte Carlo tree search program can find a winning strategy in a combinatorial game. For this purpose we propose a new combinatorial game Knight-Amazons which is a variant of the game of Amazons. Two players start the game Knight-Amazons by placing several black and white knights on the specified cells on a board. The first player selects and moves one of the white knights, and then, he or she chooses any empty cell in the range of the knight moved and thwarts it. No piece cannot be placed on the thwarted thereafter. Similarly, the second player selects and moves one of the black knights, and chooses any empty cell in the range of the knight moved and thwarts it. Two players play alternately and then the player who can no longer move any knight and thwart a cell becomes a loser of the game. And we prove that the second player at Knight-Amazons of $n \times n$ board, where n is even, can always win if he or she follows Tweedledum-Tweedledee strategy. Since there are multiple winning ways for a second player in Knight-Amazons of $n \times n$ board, where n is even, Knight-Amazons is a desired platform to study behavior of MCTS. We are interested in whether or not a MCTS program can find any winning way. We analyze the game tree of Knight-Amazons and compute win-loss ratio in several game states in a 4×4 board. Comparing the obtained win-loss ratios, we examine behavior of Monte Carlo tree search (MCTS) in Knight-Amazons. Then we execute an upper confidence bounds applied to trees (UCT) program as MCTS and find which moves the UCT program chooses most often. The result indicates that the UCT program does not necessarily converge to a Tweedledum-Tweedledee strategy nor moves having high win-loss ratio even when the number of playouts increases. It follows that a basic MCTS program cannot find a winning way such as Tweedledum-Tweedledee strategy.

Peer-review under responsibility of organizing committee of The 2015 International Conference on Soft Computing and Software Engineering (SCSE 2015)

Keywords: The game of the Amazons, Knight-Amazons, Monte Carlo tree search, Monte Carlo algorithms, UCT, Game programs, Playouts, Tweedledum-Tweedledee strategy

1. Introduction

A Monte Carlo algorithm[1] is a randomized algorithm whose output may be incorrect with a certain probability. An answer may be incorrect, however, it is much more efficient than deterministic algorithms in some cases. A *Monte Carlo tree search* (MCTS)[2,3] is one of Monte Carlo algorithms suitable for making decisions in some decision processes, most notably employed in game playing[4,5,6,7,8]. Random simulations called a *playout* in a game tree are employed to select moves in a game playing programs. A game tree is a directed graph whose nodes are states in a

* Corresponding author. Tel.: +81-18-889-2799 ; fax: +81-18-837-0406.
E-mail address: yamamura@ie.akita-u.ac.jp

Peer-review under responsibility of organizing committee of The 2015 International Conference on Soft Computing and Software Engineering (SCSE 2015)
doi:10.1016/j.procs.2015.08.406

game and whose edges are legitimate moves. MCTS is a randomized algorithm that combines the precision of tree analysis with a Monte Carlo algorithm. It has received considerable interest due to its great success in playing the game *Go*[9]. Contrary to the great success of MCTS in playing combinatorial games, it is not known whether or not MCTS can find a winning strategy that the player surely wins. One of our objectives is to verify the ability of MCTS to find out a winning strategy that the player surely wins.

We examine a MCTS by applying to a combinatorial game *Knight-Amazons* that is a variant of the *game of the Amazons*. MCTS has been applied to Amazons by many authors[6,8]. Queens at chess are used for the piece of the game in Amazons, whereas knights are used in Knight-Amazons. A knight can jump over pieces and blank cells, that is, it can move regardless of the arrangement of pieces provided that the goal cell is empty. This property of movement makes a knight quite different from the other pieces at chess. Consequently, Knight-Amazons turns out to be quite different from Amazons although the rule is almost same. As a matter of fact, there are winning strategies for the second player B to win the game in Knight-Amazons if the board size is $2k \times 2k$. The same kind of strategy does not work in Amazons. In fact, it is not known whether or not there exists any winning strategy in Amazons.

We apply the upper confidence bounds applied to trees (UCT), which is a variant of the MCTS, to Knight-Amazons in a 4×4 board. Moreover, we analyze the game tree of Knight-Amazons in a 4×4 board and then compare the results on the game tree and MCTS experiments to verify whether or not a UCT program can find a winning strategy in Knight-Amazons.

1.1. Upper confidence bounds applied to trees

Monte Carlo tree search employs a playout, which is simulation to determine the outcome of a game performed that two players play by random movement until the game ends from the initial state. The Monte Carlo tree search repeats playouts and expands the game tree and then selects the move whose win-loss ratio is maximum. An *upper confidence bounds applied to trees (UCT)* is introduced by Kocsis and Szepesvári[10] which is based on *UCB1 algorithm* proposed by Auer, Cesa-Bianchi, and Fischer[11]. A UCT algorithm selects a child state for which the UCB1 value is maximized from child states $G_1, \ldots, G_k$ of the current state G in the game tree. The $UCB1$ value of each child state G_i is defined by the following equation:

$$UCB1_i = \overline{x_i} + C\sqrt{\frac{2\log n}{n_i}},$$

where, n is the number of playouts that have been executed in the game state G, n_i is the number of playouts that are executed in the child state G_i, $\overline{x_i}$ is the win-loss ratio of G_i so far, that is, $\overline{x_j} = \frac{x_j}{n_j}$, where x_i is the number of wins in a playout in G_i. We repeat P times playouts. C is a certain constant and we do not discuss in detail here. A UCT program does not necessary have an evaluation function and its selection depends only on the result of playouts. It also searches a large game tree when the parameter P increases. Theoretically it is possible to find a winning strategy by applying an infinite number of playouts.

2. Knight-Amazons

2.1. The game of the Amazons

The game of the Amazons is a combinatorial two-player game invented in 1988 by Walter Zamkauskas[12] and first published (in Spanish) in issue 4 of the puzzle magazine El Acertijo in 1992. In Amazons we use a 10×10 board and start the game by placing four black and white queens on the specified cells on the board. The first player (W afterwords) selects and moves one of the white queens according to the move of a queen of the chess (vertical, horizontal and diagonal straight line on the board). Then the player W chooses any empty cell in the range of the queen moved and thwarts it. No piece cannot be placed on the thwarted cell nor pass through thereafter. Similarly, the second player (B afterwards) selects and moves one of the black queens according to the move of a queen, and chooses any empty cell in the range of the queen moved and thwarts it. The players W and B play alternately and then the player who can no longer move any queen and thwart a cell becomes a loser of the game. The game of the

Amazons has two characters; one is to create their own territory as the game Go, and the other is to move pieces on the board as chess. It is not easy to make an evaluation function for Amazons that is used by game playing programs to estimate the value or goodness of states because there is no value for each piece unlike chess[13]. We note that MCTS is applied to Amazons by many authors[6,8,14,15,16].

2.2. Knight-Amazons

We consider a new combinatorial game called *Knight-Amazons* that is a variant of Amazons. Pieces of knights are used in Knight-Amazons instead of queens in Amazons. At the beginning of the game, we place the same number of white knights and black knights on a $n \times n$ board in a symmetrical way. We say that position (x, y) is symmetrical to position $(n - x + 1, n - y + 1)$ on a $n \times n$ board. Therefore, if a white knight is placed at (x, y) position, then a black knight must be placed at position $(n - x + 1, n - y + 1)$ at the beginning of the game of Knight-Amazons. We do not specify the initial arrangement of pieces in this paper.

Like Amazons, the player W and the player B move their knights alternately and the player who can no longer move a knight becomes a loser of the game. Note that each player must thwart one of cells on the board that are located in the range of a knight from the position of the knight moved, otherwise he or she becomes a loser. Contrary to a queen, a knight can jump and move to any position in its range as long as it is empty. It follows that strategy of playing Knight-Amazons becomes significantly different from that of Amazons.

We use the following notation to denote a move of a knight on a board.

[(Player) : (Position of the knight selected) - (New position of knight) - (Position of thwarted cell)]

For example, [W:21-33-12] means that the player W chooses a knight at $(2, 1)$ position, moves it to $(3, 3)$ position, and thwarts the cell at $(1, 2)$ position. The move [W:21-33-12] is illustrated in Figure 1.

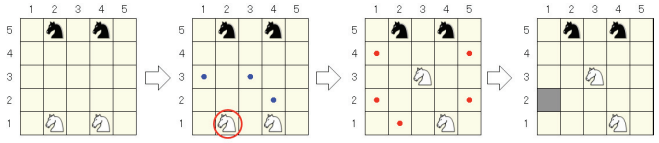

Fig. 1. Move of a white knight in Knight-Amazons [W:21-33-12]

2.3. Tweedledum-Tweedledee strategy

Tweedledum-Tweedledee strategy[17] is a strategy for the second player to win a combinatorial game by mimicking the first player. In the case of Knight-Amazons, the player B always chooses a move symmetrical to the first W's move and thwarts a cell symmetrical to the cell thwarted by the player W. If the Tweedledum-Tweedledee strategy is applicable, the second player always win because the second player always has a legitimate move after the player W plays and there is no drawn game in Knight-Amazons. For this reason, Knight-Amazons is a desired platform to study behavior of MCTS when we are interested in the ability of MCTS to find a winning strategies.

We now shall show that Tweedledum-Tweedledee strategy is applicable to the player B of Knight-Amazons in an $n \times n$ board, where n is even. A position P on an $n \times n$ board is represented as (P_x, P_y), where P_x and P_y are positive integers representing the x and y coordinates, respectively. The position symmetrical to P is $\overline{P} = (n-P_x+1, n-P_y+1)$. Suppose that the player B follows the Tweedledum-Tweedledee strategy. Whenever a white knight is placed on P, a black knight should be placed on the $\overline{P}$, and whenever P is a thwarted cell, so is $\overline{P}$. Let us call such an arrangement of an $n \times n$ board *a symmetrical point arrangement*. The initial placement of pieces in Knight-Amazons must be a symmetric point arrangement for fairness.

If P and $\overline{P}$ coincide, we have $P_x = n - P_x + 1$ and $P_y = n - P_y + 1$. Then we have $2P_x = n + 1$ and $2P_y = n + 1$. This implies that n must be odd. Therefore, if n is even, the points P and $\overline{P}$ cannot be equal. Next we shall show that the position of a thwarted cell is not the position of a knight of B nor the cell thwarted by the B's knight. Manhattan distance between P and $\overline{P}$ is equal to $|P_x-(n-P_x+1)|+|P_y-(n-P_y+1)| = 2P_x+2P_y+2n+2$. Therefore, Manhattan distance between P and $\overline{P}$ is always even.

Suppose that the game is in a symmetric point arrangement and that the player W moves a white knight in P to Q and thwarts the position R. Then a black knight is placed on $\overline{P}$. We shall show that the player B can move a black knight on $\overline{P}$ to $\overline{Q}$ and thwart $\overline{R}$. It is easy to see that if a knight moves from a position P to a position Q then Manhattan distance between P and Q is 3. On the other hand, Manhattan distance between P and $\overline{P}$ is even. Therefore, $\overline{P}$ must not be equal to Q. Thus, the player B can move a black knight on $\overline{P}$ to $\overline{Q}$.

Now we have to show the following inequalities:

$$Q \neq \overline{Q} \tag{1}$$
$$Q \neq \overline{R} \tag{2}$$
$$R \neq \overline{Q} \tag{3}$$
$$R \neq \overline{R} \tag{4}$$

If n is even then we have $T \neq \overline{T}$ for any position T as we showed above, and therefore, (1) and (4) hold. Next suppose that $Q = \overline{R}$. Since R is in the range of a knight on the position Q, Manhattan distance between R and Q is 3. On the other hand, Manhattan distance between R and $\overline{R}$ is even, which is a contradiction. Thus we have $Q \neq \overline{R}$ and so (2) holds. Similarly we can show $R \neq \overline{Q}$ and (3) holds.

We illustrate a play by B following Tweedledum-Tweedledee strategy in Figure 2.

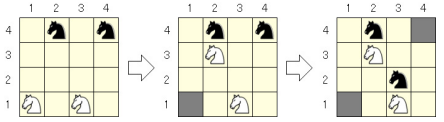

Fig. 2. Tweedledum-Tweedledee strategy in 4×4 board

Suppose that n is odd. Then $P = \overline{P}$ holds for the position $P = (\frac{n+1}{2}, \frac{n+1}{2})$. Thus, the player B cannot employ the Tweedledum-Tweedledee strategy when the player W places a white knight on the position $(\frac{n+1}{2}, \frac{n+1}{2})$. It follows that there are no obvious winning ways in Knight-Amazons using a $(2k+1) \times (2k+1)$ board.

3. Game tree analysis in Knight-Amazons

3.1. Game tree

A *game tree* is a directed graph whose nodes are states in a game and whose edges represent legitimate moves. There is no cycle in the graph of Knight-Amazons. Therefore, the states and the moves form a tree, that is, a connected graph without simple cycles. If the size of board is small enough, we can determine who is the winner of the game by recursively examining game trees provided that he or she plays correctly. If there is no child state of G, that is, there is no legitimate move at G, the player who is supposed to move a knight at G becomes a loser. We call a state with no child states an *end state*. We can completely determine who is a winner at an end state. We analyze the game tree of Knight-Amazons in a 4×4 board and we compare the results and MCTS experiments explained in subsection 4.1.

3.2. Win-loss ratio

We assume that each player moves randomly from the state G in a game tree until the game ends. In that case, win-loss ratio of each player is determined by the shape of the game tree under G. We denote the win-loss ratio of the players W and B by $R_W(G)$ and $R_B(G)$, respectively. The win-loss ratio is determined by recursively examining the child states of the game tree under G. We show how to compute the win-loss ratio $R_B(G)$ below.

If G is an end state and B is supposed to move a knight, then B becomes a loser and we define $R_B(G) = 0$. In contrast, if G is an end state and W is supposed to move a knight, then B becomes a winner and we define $R_B(G) = 1$. Now suppose that G is not an end state and B is supposed to move a knight and G has exactly k child states $G_1, \ldots, G_k$

($k \geq 1$). We define $R_B(G)$ to be the average of $R_B(G_i)$ ($i = 1, 2, \ldots, k$). Altogether we have

$$R_B(G) = \begin{cases} 0 & (G \text{ is an end state and B is supposed to play}) \\ 1 & (G \text{ is an end state and W is supposed to play}) \\ \frac{1}{k}\sum_{i=1}^{k} R_B(G_i) & (G \text{ is not an end state}) \end{cases}$$

Similarly, we define the win-loss ratio $R_W(G)$. Then we have $R_B(G) = 1 - R_W(G)$ in any state G of the game tree by the definition.

4. Experiments

4.1. MCTS Experiments

We execute a UCT program for four states $G1$, $G2$, $G3$, $G4$ of Knight-Amazons in 4×4 board each of which is obtained from the initial state by moving a white knight and thwarting a cell, and we record B's moves selected by the UCT program. the four states $G1$, $G2$, $G3$, $G4$ are illustrated in Figure 3-6. Note that the initial states of $G1$, $G2$, $G3$, $G4$ are different each other. We execute 1000 times UCT program for each of the four states. We set the parameters of the UCT program as follows: the threshold T is 1, a constant C is 1, and the number P of playouts is one of 1000, 2000, 3000, 4000, 5000. The UCT program is implemented by C# and it is executed in computer environment shown in Table 1.

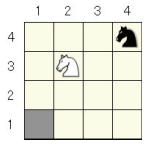

Fig. 3. $G1$

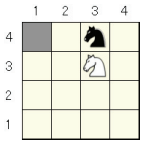

Fig. 4. $G2$

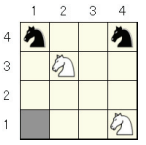

Fig. 5. $G3$

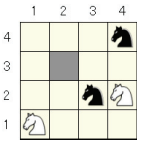

Fig. 6. $G4$

Table 1. The computer environment in experiment

OS	Windows 7 Professional(64bit)
CPU	Intel(R) Core(TM) i5(3.92GHz)
Main memory	4GB

We summarize the results of the experiments for the states $G1$, $G2$, $G3$, $G4$ in Table 2, 3, 4, 5, respectively. The number in each columns represents the number that the corresponding move is selected by the UCT program according to the number of playouts. The checkmark (✓) indicates that the corresponding move follows to the Tweedledum-Tweedledee strategy.

Table 2. Game state $G1$

move	Tweedledum-Tweedledee strategy	Number of playouts P				
		1000	2000	3000	4000	5000
B:44-32-13		0	0	0	0	0
B:44-32-24		0	0	0	0	0
B:44-32-44	✓	1000	1000	1000	1000	1000

4.2. Computation of win-loss ratio

We compute win-loss ratio of the player B in each of child states of the states $G1$, $G2$, $G3$, $G4$ given in subsection 4.1 and determine who is the winner in each case (if he or she plays correctly). We summarize the results in Table 6, 7, 8, 9, respectively.

Table 3. Game state $G2$

move	Tweedledum-Tweedledee strategy	Number of playouts P				
		1000	2000	3000	4000	5000
B:34-42-23		202	47	1	0	0
B:34-42-34		33	22	13	1	2
B:34-42-21		245	270	156	25	4
B:34-22-34		0	0	0	0	0
B:34-22-43		0	0	0	0	0
B:34-22-41	✓	60	289	656	949	990
B:34-13-34		29	29	9	3	0
B:34-13-32		196	61	0	0	0
B:34-13-21		235	282	165	22	4

Table 4. Game state $G3$

move	Tweedledum-Tweedledee strategy	Number of playouts P				
		1000	2000	3000	4000	5000
B:14-33-14		303	355	391	366	395
B:14-33-21		33	4	2	1	0
B:14-33-12		32	13	3	2	0
B:14-22-14		308	358	378	378	370
B:14-22-34		36	10	1	1	1
B:14-22-43		26	7	2	1	0
B:44-32-13		44	30	17	4	4
B:44-32-24		41	13	10	6	2
B:44-32-44	✓	177	210	196	241	228

Table 5. Game state $G4$

move	Tweedledum-Tweedledee strategy	Number of playouts P				
		1000	2000	3000	4000	5000
B:32-13-34		188	382	405	397	391
B:32-13-32	✓	585	204	171	177	165
B:32-13-21		210	414	424	426	444
B:32-24-43		0	0	0	0	0
B:32-24-32		17	0	0	0	0
B:32-24-12		0	0	0	0	0

Table 6. Game tree analysis of game state $G1$

move	Tweedledum-Tweedledee strategy	winner in $G1_i$	$R_B(G1_i)$
B:44-32-13		W	0.465
B:44-32-24		W	0.465
B:44-32-44	✓	B	0.483

4.3. Discussion

There is no winning strategy for the player B except for Tweedledum-Tweedledee strategy in each of $G1$ and $G2$ (see Table 6 and 7), on the other hand, there are winning strategies for the player B other than Tweedledum-Tweedledee strategy in $G3$ and $G4$ (see Table 8 and 9).

Table 7. Game tree analysis of game state $G2$

move	Tweedledum-Tweedledee strategy	winner in $G2_i$	$R_B(G2_i)$
B:34-42-23		W	0.519
B:34-42-34		W	0.492
B:34-42-21		W	0.551
B:34-22-34		W	0.466
B:34-22-43		W	0.466
B:34-22-41	✓	B	0.483
B:34-13-34		W	0.492
B:34-13-32		W	0.519
B:34-13-21		W	0.551

Table 8. Game tree analysis of game state $G3$

move	Tweedledum-Tweedledee strategy	winner in $G3_i$	$R_B(G3_i)$
B:14-33-14		B	0.649
B:14-33-21		B	0.592
B:14-33-12		B	0.592
B:14-22-14		B	0.649
B:14-22-34		B	0.592
B:14-22-43		B	0.592
B:44-32-13		W	0.606
B:44-32-24		W	0.606
B:44-32-44	✓	B	0.636

Table 9. Game tree analysis of game state $G4$

move	Tweedledum-Tweedledee strategy	winner in $G4_i$	$R_B(G4_i)$
B:32-13-34		B	0.515
B:32-13-32	✓	B	0.711
B:32-13-21		B	0.515
B:32-24-43		W	0.339
B:32-24-32		B	0.493
B:32-24-12		W	0.339

In the case of the states $G1$ and $G2$, the UCT program converges to choose a move following Tweedledum-Tweedledee strategy when the number of playouts increases. However, there exists a difference between $G1$ and $G2$; the UCT needs more playouts to converge in $G2$ than in $G1$. We consider this happens because the number of candidate moves in $G2$ is 9 on the other hand that in $G1$ is 3, and so candidate moves in $G2$ is comparatively bigger than that of $G1$. Moreover, win-loss ratio of Tweedledum-Tweedledee strategy is 0.483 and so lower than the other moves.

In the case of the states $G3$ and $G4$, the UCT program does not converge to any move even when the number of playouts increases. In the case of $G3$, the UCT program tends to choose the moves [B: 14-33-14], [B: 14-22-14], and [B: 44-32-44] at the same rate. The win-loss ratio of B for these three moves are higher than that of the other moves. We suspect that the UCT program tends to choose moves of higher win-loss ratio.

Contrary to the result in $G3$, in the case of $G4$ the UCT program chooses [B: 32-13-34] and [B: 32-13-21] more often than [B: 32-13-32] that follows Tweedledum-Tweedledee strategy and win-loss ratio is highest. We also admit that the the UCT program chooses the move following Tweedledum-Tweedledee strategy when the number of playouts is small whereas it tends to choose more variety of moves when the number of playouts grows.

In conclusion, the UCT program does not necessarily converge to a move following Tweedledum-Tweedledee strategy nor moves having high win-loss ratio.

5. Summary

We propose a new combinatorial game Knight-Amazons which is a variant of the game of Amazons. We rigorously proved that the second player B always wins Knight-Amazons in $n \times n$ board, where n is even, if he or she follows Tweedledum-Tweedledee strategy. This property makes Knight-Amazons a desired platform to study behavior of MCTS. Completely analyzing the game tree of Knight-Amazons in 4×4 board in several cases, we found that there are other winning ways for the player B and computed win-loss ratio of all legitimate moves. Then we compared these results with the UCT program experiments and found that the UCT program does not necessarily converge to a move following Tweedledum-Tweedledee strategy nor moves having high win-loss ratio. In some cases the UCT program does not converge to the move with the highest win-loss ration even when increasing the number of playouts. In addition, we found several game states, other than the game states we discussed here, that the UCT program does not converge to moves with the highest win-loss ration although we did not report in this paper. We did not clarify why the UCT program selects moves with lower win-loss ratio in this paper and to figure out this will be our future work.

No special processing such as pruning game trees is modified to the UCT program used in our experiment and so the program has room for improvement. Furthermore, we considered only small board of size 4×4. The size of board is too small and so we will conduct experiments of Knight-Amazons in bigger boards and study behavior of MCTS in Knight-Amazons.

References

1. R. Motwani and P. Raghavan, Randomized Algorithms, Cambridge University Press, (1995)
2. A. Bruce, The Expected-Outcome Model of Two-Player Games, Technical report, Department of Computer Science, Columbia University, (1987)
3. C.B. Browne, E.Powley, D. Whitehouse, S.M. Lucas, P.I. Cowling, P. Rohlfshagen, S. Tavener, D. Perez, S. Samothrakis, and S. Colton, A Survey of Monte Carlo Tree Search Methods, IEEE Transactions on Computational Intelligence and AI in Games, Vol. 4 (1), 1-43, (2012)
4. P. Ciancarini and G. P. Favini, Monte Carlo tree search in Kriegspiel, Artificial Intelligence, Vol. 174 (11), 670-684, (2010)
5. R. Coulom, The Monte-Carlo Revolution in Go, Japanese-French Frontiers of Science Symposium, (2008)
6. J.Kloetzer, Monte-Carlo Techniques: Applications to the Game of the Amazons, Ph.D. Thesis, JAIST, (2010)
7. C. Lee, M. Wang, G. Chaslot, J.Hoock, A. Rimmel, F. Teytaud, S. Tsai, S. Hsu, and T.Hong, The Computational Intelligence of MoGo Revealed in Taiwan's Computer Go Tournaments, IEEE Transactions on Computational Intelligence and AI in Games, Vol. 1 (1), 73-89, (2009)
8. R. J. Lorentz, Amazons Discover Monte-Carlo, CG 2008, LNCS, Vol. 5131, 13-24, (2008)
9. R. Coulom, Crazy Stone, `http://remi.coulom.free.fr/CrazyStone/`
10. L.Kocsis and C.Szepesvári, Bandit based Monte Carlo planning, 17th European Conf. on Machine Learning (ECML 2006), 282-293, (2006)
11. P. Auer, N. Cesa-Bianchi and P. Fischer, Finite-time Analysis of the Multiarmed Bandit Problem, Machine Learning, Vol. 47 (2-3), 235-256, (2002)
12. W. Zamkauskas, Amazons, `http://www.chessvariants.org/other.dir/amazons.html`
13. J. P. Neto and J. N. Silva, Mathematical Games: Abstract Games, Dover Publications, (2013)
14. J. Kloetzer, Monte-Carlo Opening Books for Amazons, CG 2010, LNCS, Vol. 6515, 124-135, (2010)
15. J. Kloetzer, H. Iida, and B. Bouzy, A Comparative Study of Solvers for Amazons Endgames, IEEE Symposium on Computational Intelligence and Games, 8033, (2008)
16. J. Kloetzer, H. Iida, and B. Bouzy, Playing Amazons Endgames, ICGA Journal, Vol. 32, (3), (2009)
17. M. H.Albert , R. J.Nowakowski and D. Wolfe, Lessons in Play An Introduction to Combinatorial Game Theory, A K Peters Ltd, (2007)